Міністерство освіти і науки України Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського” Факультет інформатики та обчислювальної техніки Кафедра інформаційних систем та технологій

Лабораторна робота №1

із дисципліни «**Технології розроблення програмного забезпечення**»

Тема **«Команди Git»**

Виконав:

студент групи ІА–24

Красношапка Р. О.

Перевірив:

Мягкий М. Ю.

Київ 2024

Мета: ознайомитися з основними командами системи контролю версій Git

Теоретичні відомості

Git — це система контролю версій, яка дозволяє розробникам відстежувати зміни в коді, співпрацювати над проектами та зберігати історію змін. Основні функції **Git** включають створення знімків (комітів) поточного стану проекту, можливість роботи з гілками для розвитку різних функціональностей незалежно одна від одної, а також злиття і вирішення конфліктів між різними версіями коду. У цій лабораторній роботі ми розглянемо основні команди **Git**, які використовуються для роботи з локальними репозиторіями, а також способи організації роботи з гілками.

Основні команди Git

git init
Створює новий локальний репозиторій у поточній директорії. Ця команда ініціалізує репозиторій і дозволяє Git почати відстеження змін.

- `git init` — створює порожній репозиторій у поточній папці.

git add
Додає зміни у файлах до індексу (стейджингу) для подальшого коміту.

- `git add <файл>` — додає конкретний файл до індексу.
- `git add .` — додає всі файли з поточної директорії.

git remove
Видаляє файл з репозиторію та, за необхідності, з робочого каталогу.

- `git rm <файл>` — видаляє файл з індексу та робочого каталогу.
- `git rm --cached <файл>` — видаляє файл з індексу, але залишає його в робочому каталозі.

git commit
Зберігає знімок стану проєкту з файлами, що були додані до індексу.

- `git commit -m "Опис змін"` — створює коміт з описом змін.
- `git commit -a -m "Опис змін"` — додає і фіксує всі змінені файли, оминувши команду `git add`.

git status
Показує інформацію про поточний стан репозиторію: які файли змінені, які готові до коміту, а які потребують додавання до індексу.

- `git status` — перегляд поточного статусу файлів у репозиторії.

git log
Виводить історію комітів, включаючи інформацію про авторів, час та повідомлення комітів.

- `git log` — перегляд історії всіх комітів.
- `git log --oneline` — скорочений перегляд історії комітів.

git branch
Показує існуючі гілки або дозволяє створити нову гілку.

- `git branch` — показує список усіх локальних гілок.
- `git branch <назва-гілки>` — створює нову гілку з поточного стану.

git checkout
Використовується для перемикання між гілками або створення нової гілки і перемикання на неї одночасно.

- `git checkout <назва-гілки>` — перемикається на існуючу гілку.
- `git checkout -b <назва-гілки>` — створює нову гілку і перемикається на неї.

git switch
Новіша команда для перемикання між гілками, яка частково замінює `git checkout`. Команда `git switch` має спрощену синтаксис для роботи з гілками.

- `git switch <назва-гілки>` — перемикається на існуючу гілку.
- `git switch -c <назва-гілки>` — створює нову гілку і перемикається на неї.

git merge
Зливає зміни з однієї гілки в іншу. Під час злиття Git намагається автоматично об'єднати зміни, але може виникнути конфлікт, якщо зміни в одному й тому ж файлі були внесені в обох гілках.

- `git merge <назва-гілки>` — зливає зміни з вказаної гілки в поточну.

git rebase
Інструмент для переписування історії комітів. Використовується для інтеграції змін з однієї гілки в іншу без створення окремого коміту злиття. Це дозволяє зберегти лінійну історію проєкту.

- `git rebase <назва-гілки>` — змінює базу поточної гілки на вказану, інтегруючи зміни.

git cherry-pick
Використовується для вибіркового перенесення окремих комітів з однієї гілки в іншу. Це корисно, коли потрібно інтегрувати певні зміни без злиття всієї гілки.

- `git cherry-pick <хеш-коміту>` — застосовує вказаний коміт до поточної гілки.

git reset
Використовується для скасування комітів або скасування змін в індексі.

- `git reset <файл>` — знімає файл зі стейджингу.
- `git reset --hard <хеш-коміту>` — повертає репозиторій до конкретного коміту, видаляючи всі зміни після нього.

Хід роботи

1. Ініціалізував порожній Git-репозиторій та створив порожній коміт

```
quiri@Roma MINGW64 ~ (branch1)
$ cd "D:\5 semester\ТРПЗ\lab1"

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1
$ git init
Initialized empty Git repository in D:/5 semester/ТРПЗ/lab1/.git/

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git commit --allow-empty -m "Init commit"
[master (root-commit) 73a6f7e] Init commit
```

2. Створив 3 гілки від master:

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git branch branch1

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git branch branch2

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git branch branch3
```

За допомогою команди *git branch* бачимо успішне виконання

```
$ git branch
  branch1
  branch2
* branch3
  master
```

3. У кожній гілці створив різну кількість комітів, відповідно до її назви.

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git switch branch1
Switched to branch 'branch1'

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1)
$ echo "Commit 1 on branch1" >> file.txt; git add file.txt; git commit -m "Commit 1 on branch1"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch1 c3e49ed] Commit 1 on branch1
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1)
$ git switch branch2
Switched to branch 'branch2'

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2)
$ echo "Commit 1 on branch2" >> file.txt; git add file.txt; git commit -m "Commit 1 on branch2"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch2 2e6e1df] Commit 1 on branch2
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2)
$ echo "Commit 2 on branch2" >> file.txt; git add file.txt; git commit -m "Commit 2 on branch2"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch2 8d6b004] Commit 2 on branch2
 1 file changed, 1 insertion(+)

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2)
$ git switch branch3
Switched to branch 'branch3'

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ echo "Commit 1 on branch3" >> file.txt; git add file.txt; git commit -m "Commit 1 on branch3"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch3 d16cd7a] Commit 1 on branch3
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ echo "Commit 2 on branch3" >> file.txt; git add file.txt; git commit -m "Commit 2 on branch3"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch3 ae933b9] Commit 2 on branch3
 1 file changed, 1 insertion(+)

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ echo "Commit 3 on branch3" >> file.txt; git add file.txt; git commit -m "Commit 3 on branch3"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[branch3 e3b812f] Commit 3 on branch3
 1 file changed, 1 insertion(+)
```

За допомогою команди *git log –oneline –all* бачимо успішне виконання

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ git log --oneline --all
e3b812f (HEAD -> branch3) Commit 3 on branch3
ae933b9 Commit 2 on branch3
d16cd7a Commit 1 on branch3
8d6b004 (branch2) Commit 2 on branch2
2e6e1df Commit 1 on branch2
c3e49ed (branch1) Commit 1 on branch1
73a6f7e (master) Init commit
```

4. Створив коміт AAA в гілці master

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ git switch master
Switched to branch 'master'

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ echo "AAA" >> file.txt; git add file.txt; git commit -m "AAA"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[master 2f4628e] AAA
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

За допомогою команди *git log –oneline* бачимо успішне виконання

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (master)
$ git log --oneline
2f4628e (HEAD -> master) AAA
73a6f7e Init commit
```

5. З гілки master роблю cherry-pick в гілку branch1

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1)
$ git cherry-pick 2f4628e
Auto-merging file.txt
CONFLICT (add/add): Merge conflict in file.txt
error: could not apply 2f4628e... AAA
hint: After resolving the conflicts, mark them with
hint: "git add/rm <pathspec>", then run
hint: "git cherry-pick --continue".
hint: You can instead skip this commit with "git cherry-pick --skip".
hint: To abort and get back to the state before "git cherry-pick",
hint: run "git cherry-pick --abort".
hint: Disable this message with "git config advice.mergeConflict false"

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1|CHERRY-PICKING)
$ notepad file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1|CHERRY-PICKING)
$ git add file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1|CHERRY-PICKING)
$ git cherry-pick --continue
Aborting commit due to empty commit message.

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1|CHERRY-PICKING)
$ git cherry-pick --continue
[branch1 1377c23] AAA
 Date: Wed Oct 2 22:44:25 2024 +0300
 1 file changed, 4 insertions(+)
```

6. З гілки master роблю merge в гілку branch2

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch1)
$ git switch branch2
Switched to branch 'branch2'

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2)
$ git merge master
Auto-merging file.txt
CONFLICT (add/add): Merge conflict in file.txt
Automatic merge failed; fix conflicts and then commit the result.

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2|MERGING)
$ git add file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch2|MERGING)
$ git merge --continue
[branch2 15f78e1] Merge branch 'master' into branch2
```

7. З гілки master роблю rebase в гілку branch3

```
quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3)
$ git rebase master
Auto-merging file.txt
CONFLICT (add/add): Merge conflict in file.txt
error: could not apply d16cd7a... Commit 1 on branch3
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply d16cd7a... Commit 1 on branch3

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ notepad file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ git rebase --continue
file.txt: needs merge
You must edit all merge conflicts and then
mark them as resolved using git add

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ notepad file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ ^C

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ git add file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 1/3)
$ git rebase --continue
hint: Waiting for your editor to close the file...       0 [sig] bash 983! sigpacket::process: Suppress
ing signal 18 to win32 process (pid 16480)
                                                    561887 [sig] bash 983! sigpacket::process: Suppressing signa
l 18 to win32 process (pid 16480)
[detached HEAD c8cf862] Commit 1 on branch3
 1 file changed, 4 insertions(+)
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply ae933b9... Commit 2 on branch3
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply ae933b9... Commit 2 on branch3

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 2/3)
$ notepad file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 2/3)
$ git add file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 2/3)
$ git rebase --continue
[detached HEAD 001e1d6] Commit 2 on branch3
 1 file changed, 2 insertions(+), 3 deletions(-)
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply e3b812f... Commit 3 on branch3
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply e3b812f... Commit 3 on branch3

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 3/3)
$ notepad file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 3/3)
$ git add file.txt

quiri@Roma MINGW64 /d/5 semester/ТРПЗ/lab1 (branch3|REBASE 3/3)
$ git rebase --continue
[detached HEAD fc50797] Commit 3 on branch3
 1 file changed, 1 insertion(+)
Successfully rebased and updated refs/heads/branch3.
```

Висновок: в ході виконання даної лабораторної роботи ми познайомились з такою системою контролю версій як Git. Ми вивчили основні команди для роботи з гітом: ініціалізація

репозиторію, додавання файлів у систему контролю, створення коміту, перенесення комітів між гілками, злиття гілок та багато інших. Окрім того ми застосували ці команди на практиці, вирішили конфлікти при переносі комітів та засвоїли вивчений матеріал.